RED-Revista de Educación a Distancia, 46(9). 15-Sept.-2015
DOI: 10.6018/red/46/9
http://www.um.es/ead/red/46/rosabel.pdf

Entornos de programación no mediados simbólicamente para el desarrollo del pensamiento computacional. Una experiencia en la formación de profesores de Informática de la Universidad Central del Ecuador

Non-symbolically mediated programming environments for the development of computational thinking. An experience in the training of computer science professors of Ecuador Central University

Hamilton Omar Pérez Narváez
Universidad Central del Ecuador. Ecuador
hopn1@alu.ua.es

Rosabel Roig-Vila. España.
Universidad de Alicante
rosabel.roig@ua.es

Resumen
Este artículo aborda la investigación, realizada con los estudiantes del primer semestre de la titulación de Informática de la Facultad de Filosofía, Letras y Ciencias de la Educación de la Universidad Central del Ecuador, cuyo propósito ha sido analizar el uso de entornos de programación no mediados simbólicamente como herramienta didáctica para el desarrollo del pensamiento computacional. Se pretende establecer las posibles ventajas de aplicar este tipo de entorno para que los estudiantes desarrollen habilidades del pensamiento computacional tales como la creatividad, modelación y abstracción, entre otras, consideradas relevantes dentro de la programación. La metodología en que se apoyó la investigación es mixta, con investigación de campo y documental a nivel descriptivo. Se utilizó como instrumento un cuestionario para la recolección de datos entre el alumnado de la titulación. Finalmente, con la información recopilada se procedió al procesamiento de datos a partir de la estadística descriptiva para, así, obtener resultados que permitiesen alcanzar las pertinentes conclusiones y recomendaciones.

Palabras clave
Pensamiento computacional, entornos de programación, habilidades computacionales.

Abstract
The present paper focuses on the research carried out with students in the first semester of the Computer Science Degree, at the Faculty of Philosophy, Letters and Education Sciences of Ecuador Central University, seeking to analyze the utilization of non-symbolically mediated programming environments as a teaching tool for the development of computational thinking. The aim sought is to identify the potential advantages of applying

the aforesaid non-symbolically mediated programming environments for students to develop computational thinking skills such as creativity, modeling and abstraction, amongst others, which are important in programming. The research was supported on a mixed methodology, with field research and documentary research at a descriptive level. A questionnaire served to collect data from the Degree students. Finally, after gathering all the information, data processing took place following descriptive statistics criteria through which conclusions and recommendations can be reached.

Keywords
Computational thinking, programming environments, computer skills.

Introducción

La introducción del uso del ordenador, por tratarse de una tecnología innovadora, ha modificado muchos procesos y actividades laborales, educativas y sociales. Ahora bien, para la mayoría de los usuarios es "invisible" el fondo tecnológico que permite su funcionamiento, tanto a nivel de hardware como de software, debido a que la mayoría de usuarios utilizan de manera instruccional esta herramienta convirtiéndose de esta manera en consumidores sin la posibilidad de explotar todas las opciones que puede brindar la tecnología existente. Para Simari (2011):

> Colocando un punto en el gráfico de la evolución que corresponda a la introducción del lenguaje escrito hace algo más de cinco mil años, y conociendo que el punto correspondiente a la imprenta se ubica en el año 1450, se reconoce que estamos en presencia de una aceleración [de un proceso que se realimenta]. Más aun cuando consideramos [que] el próximo punto lo podemos situar aproximadamente alrededor de 1940 y que representa el momento de la introducción de la computadora, cuya definición teórica publicada por Turing en 1937 corresponde a la Máquina de Turing Universal [Turing, 1937]. [Posteriormente cabe] colocar el punto que corresponde a la disponibilidad comercial de computadoras personales en los años 1970s, el correspondiente a la aparición de Internet unos pocos años después, y el de la creación de la World Wide Web (WWW) en el comienzo de los años 1990s [Berners-Lee et al. 1999] [... Está] claro observar que estamos en presencia de un proceso que se acelera exponencialmente. (p.3).

Esta aceleración exponencial se logra en base a cambios de los modelos mentales de los involucrados. Un pensamiento que resume la idea de buscar soluciones distintas a los problemas es el que expresó Zappa (s.f.) "sin desviarse de la norma, el progreso es imposible", puesto que solamente tomando nuevos caminos se produce el desarrollo del conocimiento.

Durante varios años la enseñanza de programación, en el caso de los aspirantes a docentes, de la titulación de Informática de la Facultad de Filosofía de la Universidad Central del Ecuador ha sido orientada por un modelo instruccional donde se priorizaba el aprendizaje de los lenguajes de programación, sin que los estudiantes desarrollaran una estructura de pensamiento lógico que les permitiese comprender cómo y cuándo

usar la programación en la solución de problemas. Este tipo de formación contradice lo que considera Papert (1982) como un apropiado uso del ordenador:

> En muchas escuelas de la actualidad, la frase "instrucción asistida por computadora" significa hacer que la computadora enseñe al niño. Podría decirse que se utiliza a la computadora para programar al niño. En mi concepción, el niño programa a la computadora y, al hacerlo, adquiere un sentido de dominio sobre un elemento de la tecnología más moderna y poderosa y, a la vez, establece un íntimo contacto con algunas de las ideas más profundas de las ciencias, la matemática y el arte de la construcción de modelos intelectuales. (p. 17-18).

Es una labor difícil de alcanzar con los estudiantes universitarios en programación la propuesta de Papert debido a que la metodología de trabajo instruccional y los recursos tecnológicos educativos aplicados impiden desarrollar apropiadamente varias habilidades propias de la informática y de programación como abstracción y creatividad, entre otras. Además, tratando de realizar una comparación con lo que menciona el autor y la realidad de esta población seleccionada para estudio, si bien existe una diferencia de edad cognitiva con los niños, principalmente se evidencia que también a este nivel sigue existiendo esta dificultad en el momento de poner en juego procesos mentales superiores para resolver problemas debido a variadas circunstancias. Una de las posibles causas que generan esta situación está en la escasa comprensión del problema debido a que "la primera etapa en la resolución de un problema es la comprensión acabada del mismo" (Señas y Moroni, 2002, p.3). Concuerdan con esta idea lo que expresan Vega y Espinel (2010) acerca de la manera en que se pueden resolver problemas de programación, indicando que "es conveniente que el estudiante desde el primer momento conozca la importancia de esta área del conocimiento como un medio para la solución de problemas" (p.8). Asimismo, señalan la necesidad de alcanzar en el estudiante la comprensión del uso de la tecnología informática

> no como un simple requisito para aprender un lenguaje de programación mediante la elaboración de algoritmos y la generación de códigos, sin previamente haber entendido el problema que se pretende resolver y sin un proceso metodológico adecuado, del cual se aprende a programar básicamente pero no se soluciona el problema. (p.8).

Un segundo factor que puede impedir el desarrollo del pensamiento lógico por el cual abogamos corresponde a migrar del lenguaje natural a un lenguaje artificial como es el de programación, utilizando para ello principalmente software de aplicación diseñado por grandes corporaciones que se manifiesta en actividades concretas con estos programas. Un ejemplo sería el siguiente: usar un compilador de una empresa específica obliga a escribir ciertos códigos a su manera o limita el uso de otros; de esta forma, se aprende directamente a programar sobre el lenguaje, pero sin orientar al estudiante hacia la creatividad, la resolución de problemas y el razonamiento abstracto lógico. En una investigación que realizan Ferreira y Rojo (2006) manifiestan esta situación de la siguiente manera:

> Algunos enseñan a programar en un lenguaje de programación particular, utilizando su sintaxis y su semántica, y otros emplean un lenguaje algorítmico lo bastante

> general como para permitir su traducción posterior a cualquier lenguaje de programación. El primero de estos enfoques tiene el inconveniente de ligar los conceptos básicos de la programación a un lenguaje determinado, el cual tiene sus propias características y especificidades, perdiendo de vista los conceptos generales. Esto lleva a que existan varios cursos de enseñanza de programación que no son otra cosa más que la enseñanza de los conceptos básicos pero en otros tantos lenguajes, con otras características y con otras especificidades. Así podemos encontrar en una currícula cursos de Pascal, C, FORTRAN, Ada, Clipper, C++, Java, Delphi, VisualBasic, etc. (p.101).

En cierta manera, al trabajar directamente sobre el lenguaje de programación, el desarrollo de la lógica de programación se ve relegado por la importancia que cobra la escritura correcta de las sentencias y el manejo de la estructura propia de cada lenguaje. Como expresa Taborda (2012):

> Los lenguajes naturales tienden a tener una estructura declarativa, mientras que los lenguajes de programación una estructura imperativa, haciendo de la primera una estructura más ambigua. La diferencia, sin embargo, es especialmente marcada en la semántica de los conectores lógicos; así, el conector *entonces* es entendido en programación como *en esta condición*, en lugar de *después* como ocurre en el lenguaje natural. (p.4).

Esto acaba por ser un problema para el estudiante, puesto que el significado de las instrucciones utilizadas en programación, al ser distinto del que cotidianamente emplea, complica su comprensión y manifiesta una dificultad en el rendimiento académico. Así, según indica Pérez (2011) en un estudio sobre programación en la titulación de Informática de la Universidad Central del Ecuador, "se evidenció que la problemática estaba presente principalmente en los años iniciales, donde los promedios son inferiores a la nota mínima de promoción y revisando los siguientes datos se establece que el nivel de deserción y repetición es alto con un porcentaje del 58,5%"(p.6). Estos problemas siguen manteniéndose hasta la actualidad y, de ahí, la necesidad de proponer estrategias de solución que sean innovadoras y, a la vez, sirvan positivamente para mejorar la calidad de los aprendizajes. Es necesario, para ello, apoyarse en herramientas digitales motivadoras con entornos amigables de uso para el estudiante que permitan también proponer nuevos tipos de problemas y retos que vayan más allá de los problemas tradicionales y busquen la posibilidad de combinarse con distintas áreas o disciplinas del conocimiento.

Esta idea acerca del uso de los ordenadores y su finalidad es coherente con lo que expresa Larrea (2013) en una visión general acerca del currículo de la educación superior ecuatoriana y del cual la formación de los estudiantes de la titulación de Informática no puede dejar de tomar como referente, ya que actualmente se halla inmersa en procesos de acreditación y rediseño bajo esta perspectiva:

> Los nuevos modelos académicos de la educación superior deben considerar los cambios que se operan en los horizontes epistemológicos del conocimiento, las nuevas tendencias de la educación superior a nivel latinoamericano y mundial, las

> reformas académicas, normativas, perspectivas y planes de desarrollo, visiones y necesidades de los actores y sectores, si queremos hacer de las Instituciones de Educación Superior instituciones pertinentes y de calidad. (p. 2).

Las ideas antes mencionadas acerca de habilidades computacionales, formas de enseñanza basadas en la resolución de problemas de múltiples proceso y lógica de programación, entre otros, se hallan recogidos en lo que denominamos pensamiento computacional. Según Wing (2009b), autora que acuñó dicho término:

> El pensamiento computacional será una habilidad fundamental utilizada por todos en el mundo. A la lectura, escritura y aritmética, vamos a añadir el pensamiento computacional a la capacidad de análisis de cada niño. El pensamiento computacional es un enfoque para la solución de problemas, construcción de sistemas, y la comprensión del comportamiento humano que se basa en el poder y los límites de la computación. Si bien el pensamiento computacional ya ha comenzado a influir en muchas disciplinas, desde las ciencias a las humanidades, lo mejor está aún por venir. De cara al futuro, podemos anticipar incluso efectos más profundos del pensamiento computacional en la ciencia, la tecnología y la sociedad. Entretanto, habrá nuevos descubrimientos e innovación y las culturas evolucionarán. (pp.35-36).

Cabe decir que el pensamiento computacional está relacionado también con otros tipos de pensamiento, como el matemático, el lógico y el crítico, con los cuales también participa, puesto que comparte habilidades cognitivas comunes tales como el reconocimiento de patrones, abstracción, modelación y otras más. Su finalidad es que a partir del reconocimiento de los aspectos que nos rodean, de problemas reales de las actividades diarias o de las ciencias, propongamos soluciones aplicando herramientas informáticas.

Si combinamos las potencialidades del ser humano con las de la máquina, esto permitirá alcanzar nuevos niveles, tanto en la solución de problemas como en el desarrollo cognitivo y actitudinal ante los nuevos retos para la humanidad que se presentan en el siglo XXI (Voogt, Fisser, Good, Mishra y Yadav, 2015). Desde esta perspectiva se han postulado diversos autores con el fin de introducir el pensamiento computacional en tempranas edades y desde diversas disciplinas. Así, Barr, Harrison y Conery (2011), exponen los planteamientos generales respecto al alumnado K-12 en el marco que ofrece la *International Society for Technology in Education* (ISTE) (ver https://goo.gl/l72AWS). En el ámbito universitario, Anthony y Soon (2015) nos muestran una experiencia en la cual se detalla el trabajo realizado por el alumnado en torno al diseño de una plataforma cuya finalidad es el diseño de juegos y el aprendizaje social a partir del pensamiento computacional. Por otro lado, Jin y Zhaohui (2015) abordan también en el ámbito universitario el desarrollo del pensamiento computacional a través del modelo didáctico *Inquiry Teaching Model*; o Park, Song y Kim (2015), los cuales implementan una experiencia educativa para contrastar resultados académicos resultantes de un proceso basado en el pensamiento computacional.

La mayoría de personas desaprovecha las potencialidades de un ordenador porque desconoce los conceptos que permiten usarlo de mejor forma y con más creatividad y explotan poco su capacidad de apoyo como herramienta para el desarrollo del conocimiento. Así, la mayoría de usuarios utilizan fundamentalmente el ordenador para realizar trabajos de oficina y actividades de diversión.
Al respecto, Zapotecalt (2014) manifiesta que:

> A diferencia de otras disciplinas del conocimiento, un especialista de las ciencias de la computación confronta problemas y propone soluciones en cualquier área del conocimiento. Las habilidades de un especialista en computación son más que ser capaz de manipular diversas aplicaciones de oficina, o de programar una computadora. Un especialista en computación crea y hace uso de diferentes niveles de abstracción, para entender y resolver problemas con mayor eficacia. Jeannette Wing sostiene que la esencia del pensamiento computacional es la abstracción y que las abstracciones para la computación son "las herramientas mentales " y las computadoras las herramientas "metálicas" que automatizan las abstracciones. (p.9).

A nivel mundial existen múltiples iniciativas que impulsan la enseñanza de programación. Una de las más notorias es la iniciativa Code.org que utiliza diferentes herramientas tecnológicas como medio de apoyo para la enseñanza de programación, entre ellas la herramienta Scratch (Olabe, Olabe, Basogain, Maiz y Castaño, 2011) y LighBot. En Estonia, la fundación "Tiger Leap Foundation" impulsa desde septiembre de 2012 un programa denominado "Progre Tiger", que alienta el aprendizaje de programación y creación de aplicaciones web en la etapa de educación inicial. Sus principales objetivos son: desarrollar entre los jóvenes el pensamiento computacional, la creatividad y las habilidades matemáticas; demostrar que "la programación puede ser interesante, enseñar los fundamentos de la programación a través de la actividad práctica y enseñar a los alumnos a utilizar diferentes lenguajes de programación adecuadas a la edad." (Banchoff, Díaz, Queiruga, y Martín, 2014, pág. 5)

En los países de América, especialmente en EE.UU., la asignatura de Programación está integrada dentro de la cultura educativa. Usar un lenguaje de programación como soporte a las Matemáticas, Física o Química, o incluso áreas que no son exclusivas de Ciencias, es algo habitual en muchas escuelas de secundaria o las titulaciones universitarias. Muchos centros educativos en esos países trabajan con herramientas que ayudan a una mejor didáctica de la Programación, alejándose de habituales entornos y lenguajes profesionales de Programación (Carralero, 2011, pág. 1).

Una de las alternativas destinadas a que los alumnos aprendan programación de manera fácil es la utilización de herramientas tecnológicas con el objetivo de crear entornos multimedia, juegos y animaciones que les permitan desarrollar el pensamiento computacional, huyendo de aburridos entornos de consola. Cuando Jannette Wing utilizó el término pensamiento computacional para referirse a la actividad mental de formular y resolver problemas que admitan soluciones computacionales (Wing, 2009a), esta nueva noción fue unida a nuevos lenguajes de programación, tales como Scratch, que forma parte "de un movimiento que aboga actualmente porque cada estudiante, de

cada escuela, tenga la oportunidad de aprender a programar" (López, 2014, p. 9). Desde la fecha en que se abrió el sitio web de Scratch se han subido más de 2.4 millones de proyectos elaborados por más de trescientos mil usuarios de diferentes partes del mundo, principalmente jóvenes entre los 11 y 18 años de edad. Según los administradores del sitio, mensualmente se sube un promedio de cincuenta mil proyectos y se realizan cerca de trescientos mil comentarios.

La aparición de programas con entornos de programación mediados simbólicamente que buscan facilitar el aprendizaje de programación no es del todo nueva como se podría pensar. Revisando literatura acerca del tema, encontraremos referencias acerca de LOGO y Turtle Arts como predecesores del movimiento que Scratch, Alice o Visual Da Vinci están promoviendo y al que hemos llamado con ese nombre por sus características de interfaz amigable que, además, posibilita el uso y creación de recursos multimedia, los cuales no necesariamente poseen los compiladores usados convencionalmente en la programación orientada a objetos de lenguajes como Java o C++.

Nos referimos, pues, con el término "entornos de programación mediados simbólicamente", del que posiblemente no exista precedente en la literatura, a aquellos entornos de programación donde existe una interface conceptual entre el lenguaje de programación propiamente dicho y el programador. Son las órdenes, la sintaxis, los comandos, los procedimientos, las variables, entre otros, los que se muestran gráficamente, liberando, así, al usuario del problema de usar una sintaxis rigurosa y a veces complicada, y centrando su potencial en la resolución de un problema de forma algorítmica.

Esta interfaz con la que se construye el código y que está constituida por elementos visuales de tipo simbólico como pueden ser los que rodean a la tortuga, en el caso de LOGO y de Turtle Arts, o el gato, en el caso de Scracht, ha sido considerada en los últimos años a nivel mundial como una alternativa para el aprendizaje de programación en niños, jóvenes y adultos, así como para el desarrollo de las habilidades propias del pensamiento computacional.

Por ejemplo, en Colombia se promueve "el uso de Scratch en los Tecno-parques, como parte de la formación en programación y creatividad de los jóvenes. La Fundación Compartir en los Computer Clubhouse de Bogotá y la Fundación Telefónica, mediante el portal Educared, divulga y facilita talleres de formación virtual sobre Scratch" (Ospina, 2012). Según indica este mismo autor, la Academia Nacional de Telecentros ha ofertado cursos de creatividad digital usando este software durante diversos ciclos de formación de coordinadores de Telecentros.

En 2013 la Fundación Gabriel Piedrahita Uribe (FGPU) publica el resultado del proyecto de investigación "Programación de computadores y desarrollo de habilidades de pensamiento en niños escolares: fase exploratoria", que fue ejecutado por los investigadores Taborda y Medina (2013). En el informe final se destaca que:

> Los resultados del análisis de tareas muestran en detalle la forma como el uso del entorno gráfico de programación Scratch, junto con las actividades educativas

> propuestas en el aula, promueven el desarrollo del pensamiento computacional, la adquisición de conocimiento conceptual académico y habilidades de planificación cognitiva. (p.17)

En 2006 Ecuador empieza a incorporar formalmente las TIC dentro de los diferentes sectores. "En el sector educativo del país se ha apuntado a la dotación de infraestructuras, equipamiento de aulas con ordenadores y recursos informáticos, dotación de software educativo, capacitación al profesorado, creación de portales educativos, soporte técnico a las escuelas, entre otros" (Peñaherrera, 2012, p. 2). Ahora bien, son escasos los proyectos desarrollados basados en herramientas tecnológicas del tipo Scratch para desarrollar el pensamiento computacional. Únicamente se ha llevado a cabo un proyecto implementado por la Escuela Politécnica del Litoral (ESPOL) a nivel de educación básica y otro por la Universidad Yachay en colaboración con Clear mind-it organizado con niños de entre 8 a 12 años, pero no hay evidencias de su uso en la educación superior y los resultados logrados con el uso de la herramienta, por lo que se hace necesario investigar acerca del tema.

A partir de las consideraciones anteriores y la casi inexistencia de este tipo de proyectos en Ecuador, nace la propuesta de realizar una investigación cuyo objetivo general fue examinar el uso de un entorno de programación no mediado simbólicamente como herramienta didáctica para el desarrollo del pensamiento computacional con los estudiantes del primer semestre de los estudios de Informática de la Facultad de Filosofía de la Universidad Central del Ecuador.

En la formación de profesionales docentes en informática de la Facultad de Filosofía de la Universidad Central del Ecuador no existen investigaciones anteriores relacionadas con el diagnóstico de habilidades del pensamiento computacional tales como creatividad, lógica del pensamiento o trabajo colaborativo, ni tampoco sobre las características de los recursos de software utilizados en el proceso de enseñanza y aprendizaje, o sobre la opinión de los estudiantes en relación al entorno, recursos que presta y acceso a dicho proceso. Esta falta de información no permite establecer una adecuada valoración de resultados alcanzados en elementos propios del pensamiento computacional como es la creatividad, que, en resumen, Zapata (2014) expresa como el resultado de combinar varios tipos de pensamientos, como el convergente y el divergente, que aun pareciendo dos formas distintas de pensamiento, se combinan y orientan hacia la generación de ideas realizando asociaciones inusuales basadas en la indagación y fluidez del pensamiento. Consideramos que esta habilidad es importante en cualquier profesión, y más en el docente, que debe poseerla a la hora de enfrentarse, en primer lugar, a los problemas de su aprendizaje y posteriormente en la forma en que enseñará a sus estudiantes a desarrollarla.

Además, otra de las habilidades importantes para indagar cómo se desarrolla es el trabajo colaborativo. A decir de Guerra (2008), refiriéndose a este tipo de aprendizaje, "se sustenta en dos teorías de aprendizaje que se complementan: la Teoría de Aprendizaje Social, Bandura (1989), y la Teoría de la Actividad (…) está siendo usada como marco teórico para entender los aspectos sociales del aprendizaje asistido por computadoras" (p.14). De hecho, al ser la titulación de Informática una carrera

universitaria relacionada directamente con el uso de la tecnología de la información, es importante saber si la metodología, recursos y otros elementos relacionados con el aprendizaje son los apropiados en relación con los resultados esperados. Además, probablemente los mismos también serán aplicados por los egresados en su vida profesional, de ahí su importancia para ser analizados.

La investigación, pues, se centró en los siguientes objetivos específicos:

- Indagar las ventajas que proporciona el uso de Scratch en el aprendizaje de programación de los estudiantes de la titulación de Informática de la Facultad de Filosofía.
- Examinar la preferencia de los estudiantes por el uso de entornos de programación mediados en relación con el aprendizaje colaborativo de programación y la creatividad como habilidades del pensamiento computacional.
- Identificar el nivel de incursión de los estudiantes de la titulación de Informática en el desarrollo de proyectos Scratch.

Metodología

El diseño de la investigación fue mixta, con enfoques cualitativo y cuantitativo, de nivel descriptivo, no experimental, de campo con apoyo documental. La población seleccionada fue de 35 estudiantes de ambos géneros del primer semestre del curso 2014-15 de la titulación de Informática de la Facultad de Filosofía de la Universidad Central del Ecuador.

Con el grupo mencionado, que correspondió a un curso, y durante dos meses en sesiones de tres horas semanales se trabajó una Unidad Didáctica referente a algoritmos utilizando Scratch basándose en la metodología de resolución de problemas y el ciclo del aprendizaje de Kolb. Las sesiones de trabajo se planificaron de la siguiente manera:

- Primera sesión, correspondiente a temas de introducción a los algoritmos: se trabajan cinco ejercicios con Scratch que manejen sentencias de ingreso y salida de información.
- Segunda sesión: se desarrollan temas referentes operaciones matemáticas, movimientos y gráficos.
- Tercera sesión: se trabajan sentencias condicionales mediante la resolución de problemas y diseño de juegos interactivos.
- Cuarta sesión: se trabajan condiciones compuestas aplicando operadores lógicos.
- Quinta sesión: se desarrolla el tema de lazos de repetición, o también llamados bucles, donde se propone elaborar proyectos que incluyan todas las sentencias aprendidas.
- Sexta sesión: se trabajan los lazos condicionales como *mientras* y *repetir mientras*.
- Séptima sesión: se realiza un trabajo con las herramientas multimedia que facilita Scratch.

Se presentaron ejercicios resueltos como parte del proceso didáctico, así como vídeos acerca de proyectos realizados. Posteriormente se procedió a plantear problemas para ser resueltos individual y grupalmente que fueron evaluados por el docente durante cada clase. Estos proyectos han sido publicados en el sitio de internet que provee Scratch (www.scrathc.mit.edu) en las cuentas individuales de los estudiantes. En ciertos momentos también se sugirió al estudiante que aplicase sus conocimientos en proyectos libres que denotasen creatividad y soluciones a problemas concretos. Al finalizar todo el proceso se aplicó una encuesta con un instrumento tipo cuestionario diseñado como escala de satisfacción.

Es necesario indicar que la mayor parte de estudiantes no habían tenido una experiencia previa con el uso de la herramienta informática Scratch, así como también con el trabajo de algoritmos para el diseño de programas, puesto que muchos de ellos son graduados en el Bachillerato General Unificado en Ciencias y un escaso número en el Bachillerato Técnico en Informática.

Instrumento para recolección de información

El cuestionario estuvo compuesto por 23 preguntas con escala tipo Likert impar (ver anexo1), con opciones que van en posibilidades de *siempre, casi siempre, a veces, casi nunca* y *nunca*. Los ítems se elaboraron a partir de los objetivos y, con el fin de operacionalizar la recolección de información, se tuvieron en cuenta las siguientes dimensiones:

- Interfaz con indicadores acerca de recursos y acceso.
- Habilidades del pensamiento computacional con indicadores acerca del trabajo colaborativo, creatividad, desarrollo del pensamiento elaboración de juegos, presentaciones animadas y proyectos.
- Metodología de enseñanza que tuviera en cuenta indicadores tales como métodos activos, pasivos, individuales y colaborativos.

A fin de certificar la validez del instrumento se solicitó a tres docentes de la propia titulación, expertos en informática y educación, que lo analizasen en referencia a tres aspectos: calidad técnica, relación con objetivos y lenguaje. Las observaciones realizadas fueron tenidas en cuenta antes de aplicar el pilotaje.
Posteriormente se realizó un pilotaje con un grupo similar de estudiantes de la titulación de Informática de segundo semestre constituido por 36 estudiantes que ya habían trabajado el entorno de programación en el ciclo anterior en la misma asignatura que la población seleccionada para la investigación.
Con los datos recogidos se sometió a un análisis de confiabilidad mediante el alfa de Cronbach el instrumento. Se obtuvo un valor de 0,84, por lo que puede ser considerado altamente confiable de acuerdo a la escala de George y Mallery.

Tabla N.°1

Estadísticos de fiabilidad

Alfa de Cronbach	Alfa de Cronbach basada en los elementos tipificados	N de elementos
,840	,835	36

De forma esquemática, el proceso realizado durante la investigación fue el siguiente:

- Investigación documental previa
- Determinación de objetivos.
- Elaboración de instrumentos.
- Validación de expertos.
- Pilotaje con los instrumentos.
- Análisis de confiabilidad del instrumento.
- Aplicación en la población.
- Análisis de información apoyado en estadística descriptiva y la herramienta SPSS.
- Elaboración de conclusiones y recomendaciones.

Resultados

Entre los principales hallazgos de la investigación cabe destacar los siguientes:

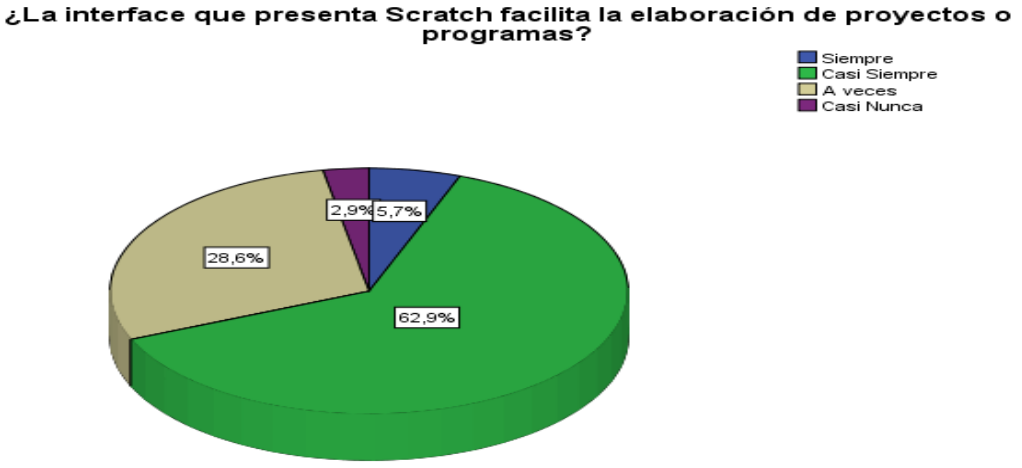

Gráfico N° 1. Fuente: Autor

Tal y como se aprecia en el gráfico 1, un alto porcentaje de estudiantes consideran que la interface de Scratch facilita la elaboración de proyectos o programas, lo cual puede deberse a las herramientas multimedia que se proporcionan en el mismo.

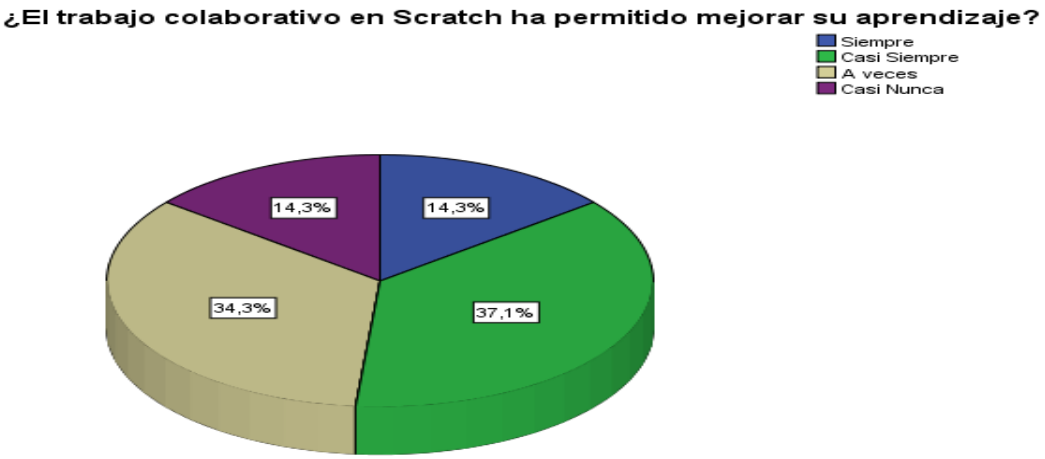

Gráfico N°2 Fuente: Autor.

Los estudiantes manifiestan en un porcentaje importante que casi siempre el trabajo colaborativo en Scratch mejora su aprendizaje; lo que podría entenderse como aprobación al uso de la herramienta informática.

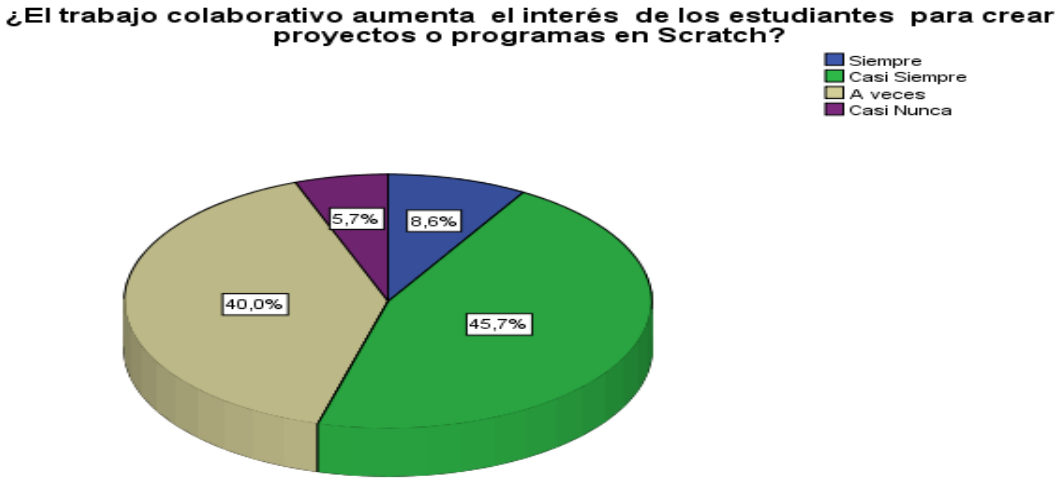

Gráfico N°3. Fuente: Autor

Los entrevistados manifiestan en altos porcentajes que casi siempre o a veces el trabajo colaborativo del entorno Scratch aumenta su interés por aprender programación (ver gráfico 3) que se manifiesta en la creación de proyectos.

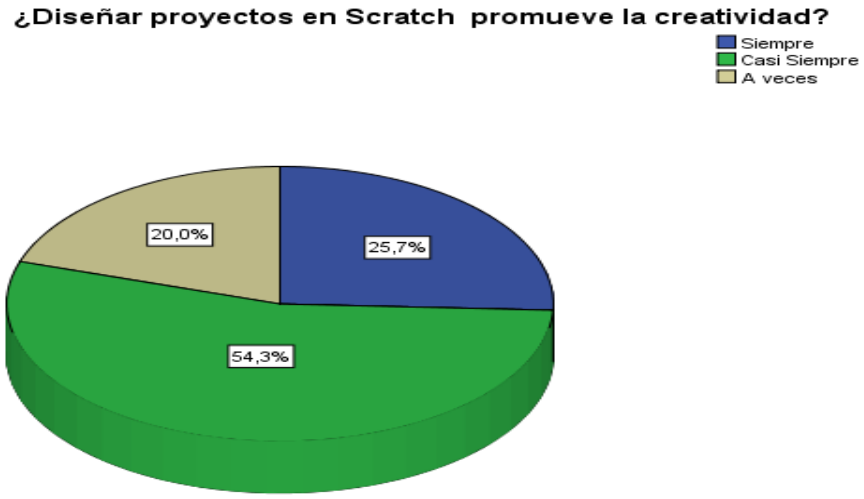

Gráfico N° 4. Fuente: Autor.

Al ser consultados acerca el nivel de creatividad que promueve la herramienta tecnológica, el alumnado manifiesta en un porcentaje mayoritario (ver gráfico 4) que efectivamente diseñar proyectos en Scratch promueve la creatividad.

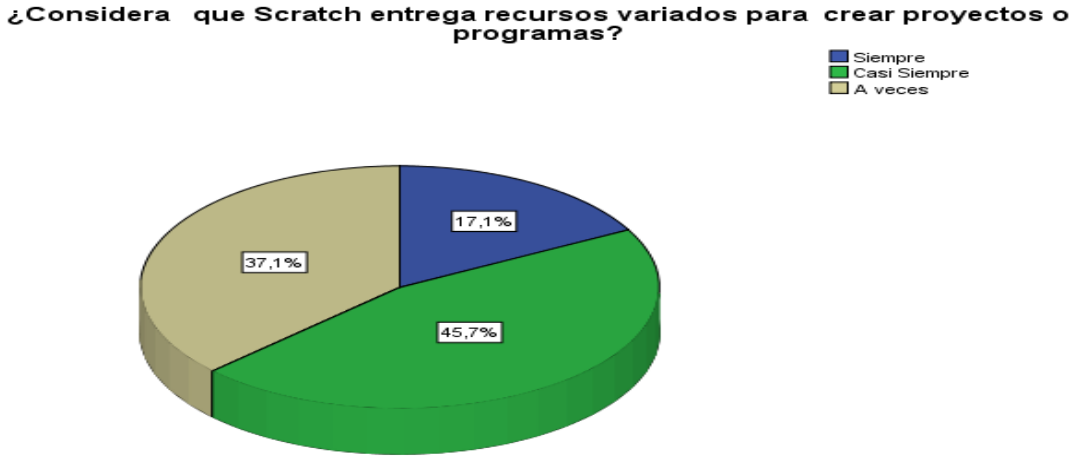

Gráfico N° 5. Fuente: Autor

En cuanto a la ventaja de poseer recursos variados para la creación de proyectos o programas, el alumnado manifiesta en un alto porcentaje (ver gráfico 5) que casi siempre esto sucede al utilizar la herramienta informática.

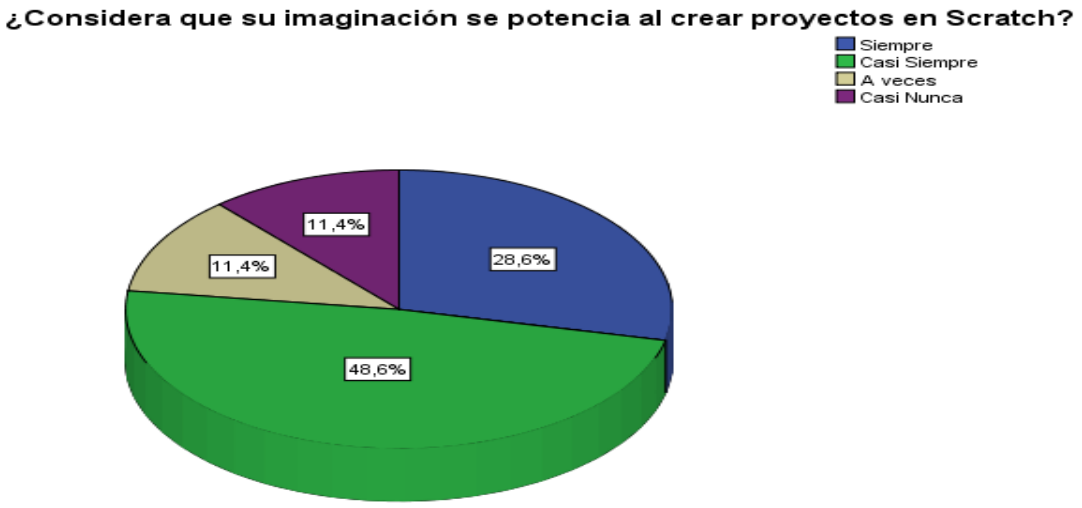

Gráfico N° 6. Fuente: Autor

En referencia a la imaginación que ponen al elaborar proyectos en Scratch (ver gráfico 6), un 48,6% responde que casi siempre se ve potenciada, lo cual es un alto porcentaje.

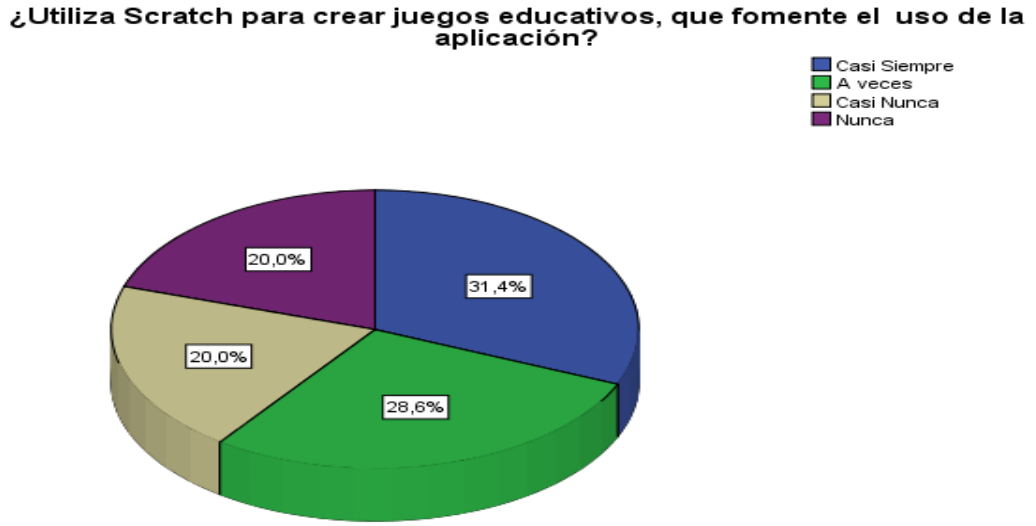

Gráfico N° 7. Fuente: Autor

La respuesta ante la pregunta sobre si los estudiantes han creado juegos en Scratch con la intención de fomentar el uso de la aplicación por otras personas, esta se muestra dispersa (ver gráfico 7) y el mayor porcentaje manifiesta que a veces lo ha hecho (28,6%).

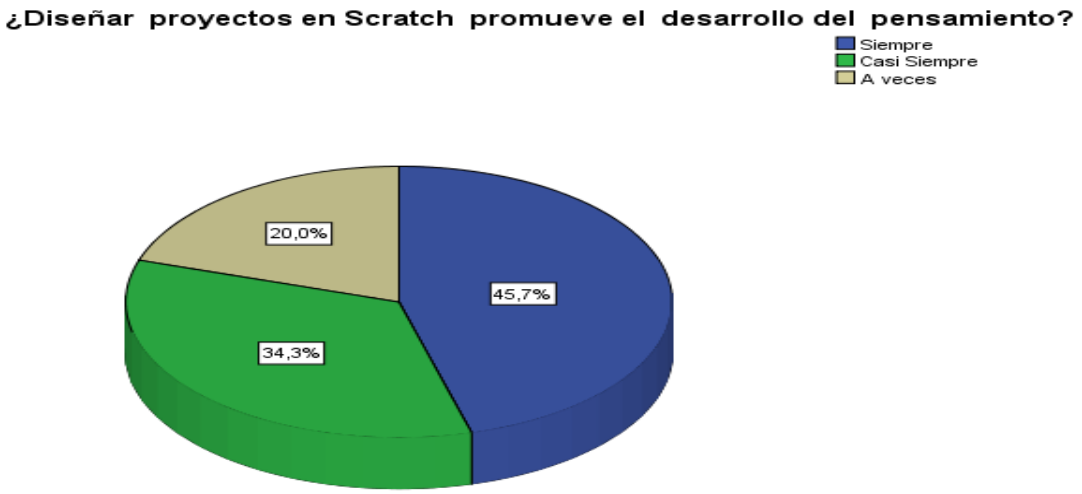

Gráfico N° 8. Fuente: Autor

En referencia a si consideran que diseñar proyectos en Scratch promueve el desarrollo del pensamiento (ver gráfico 8), existen altos porcentajes que están de acuerdo con esta aseveración (un 45,7% contestan que siempre y un 34,3% que casi siempre).

Discusión y conclusiones

Nos habíamos propuesto como objetivo indagar las ventajas que proporciona el uso de Scratch en el aprendizaje de programación de la población referida. En este sentido, una de las ventajas según los estudiantes está relacionada con el entorno de programación que ofrece la herramienta, ya que al ser de tipo visual y sin la complicación de sintaxis de otras herramientas, facilita la elaboración de proyectos o programas, facilita el modelado de objetos, así como la abstracción de los mismos. Si se compara con una experiencia realizada por Taborda y Medina (2013) se encuentran similitudes que se manifiestan en su investigación de la siguiente manera:

> El programa ofrece un claro soporte para algunos de los elementos que en la literatura sobre aprendizaje de la programación se han señalado como los más problemáticos para los aprendices [...] tales como el uso de iteraciones y de condicionales. [...] Estas funciones en específico se facilitan mediante el uso de bloques prediseñados que sirven de marco para construir un programa correctamente. (p.17).

En el caso de la población seleccionada de la titulación de Informática de la Facultad de Filosofía de la Universidad Central, un 62,9% expresa que la interfaz facilita la elaboración de proyectos y un 45% manifiesta que casi siempre Scratch ofrece recursos variados para estos proyectos. En resumen, la opinión de los estudiantes es favorable mayoritariamente con respecto a usar el programa para aprender programación.

Por otro lado, cabe decir que en referencia a mejorar el aprendizaje de programación mediante la herramienta digital Scratch, son pocos los estudiantes que expresan una opinión desfavorable sobre el aprendizaje con dicha herramienta, lo que se puede entender como una aceptación positiva al uso de la misma y sus ventajas para el aprendizaje de programación. En relación a esta conclusión, Taborda y Medina (2013) concuerda con las ventajas que puede ofrecer la herramienta informática y su entorno de trabajo para los estudiantes al expresar:

> el uso de Scratch puede tener impacto en campos de conocimiento que si bien se circunscriben al pensamiento computacional, no se limitan al aprendizaje de la programación. Así, es importante destacar la contribución que hace el uso del entorno de programación para aprender cómo modelar la realidad en términos de variables que interactúan y así promover un pensamiento más abstracto. (p.18.).

Así pues, se puede concluir una opinión favorable probablemente debido a la interfaz suficientemente intuitiva y gráfica basada en trabajo con objetos, escenarios y acciones que facilita al programador el trabajo.

El segundo objetivo que nos habíamos planteado hacía referencia a examinar la preferencia de los estudiantes por el uso de entornos de programación mediados en relación con el aprendizaje colaborativo de programación y la creatividad como habilidades del pensamiento computacional. Pues bien, cabe decir que se expresa una opinión favorable de los estudiantes acerca del trabajo colaborativo que promueve Scratch en relación con los aprendizajes alcanzados, puesto que comparten información en sitios de la red, revisan vídeos o se apoyan en las opiniones de sus compañeros acerca de cómo usar las instrucciones. Este resultado refuerza las conclusiones a las que llegan en su investigación Brennan y Resnick (2012) sobre la evaluación del pensamiento computacional mediante el entorno mediado de programación. Tal y como indica este autor, "en la comunidad en línea de Scratch, la retroalimentación y crítica de los pares se valora mucho. La evaluación debe acoger esta multiplicidad de puntos de vista, involucrando, como posibles y apropiadas, las evaluaciones de la persona, los pares, los padres, los maestros y los investigadores." (p.27). Los estudiantes de la muestra, en un porcentaje del 37,1% opinan que casi siempre los entornos de programación mediados no simbólicamente apoyados en el trabajo colaborativo mejoran su aprendizaje. Así, también manifiestan en un 45% que su interés por desarrollar proyectos Scratch aumenta si trabajan colaborativamente. Sin embargo, con el estudio realizado cabe decir que los resultados no son completamente confirmatorios sobre las ventajas que manifiesta el software Scratch en el desarrollo del aprendizaje colaborativo.

Con respecto a la creatividad, un alto porcentaje de estudiantes expresa que el trabajo con Scratch, a partir de los recursos que proporciona y las posibilidades que posee la herramienta, promueve dicha habilidad. Al mismo tiempo, manifiestan que ven potencializada su imaginación y el pensamiento computacional. Coincidimos, así, con la experiencia realizada por los investigadores Bustillo, Vizcarra y Aristizabal (2014) al manifestar las ventajas con la herramienta Scratch, así como la percepción de sentirse

más capaces, esto a pesar de las diferencias de conocimientos y heterogeneidad de ambos grupos. A nivel mundial se habla de cerca de un millón de proyectos compartidos en línea que pueden ser reusados y modificados. Se pudo observar en los estudiantes durante el tiempo de uso de la herramienta sugerir cambios, aplicar nuevas funciones, buscar nuevas soluciones y crear sus propios escenarios u objetos. Es un alto porcentaje de los estudiantes que manifiestan que casi siempre (54,3%) ven un potencial creativo al diseñar proyectos en Scratch y reconocen en un 48,6% que su imaginación también se potencia.

El último objetivo que nos habíamos planteado era identificar el nivel de incursión de los estudiantes de la titulación de Informática en el desarrollo de proyectos Scratch. En este sentido cabe decir que este nivel ha sido inferior en comparación al que, por ejemplo, refiere Resnik (2007, p. 4): "con Scratch, los estudiantes cambian de consumidores de medios a creadores de medios, creando sus propias historias interactivas, juegos y animaciones y luego compartiendo sus creaciones en la Web." El grupo investigado muestra escasa incursión en este tipo de actividad. Este punto se convierte en una tarea pendiente de los estudiantes y docentes de la titulación de Informática de la Universidad Central del Ecuador. Es importante que el profesor motive a sus estudiantes a proponerse nuevos retos y los estudiantes, a su vez, diseñen sus propios proyectos y los compartan en la red en busca de opiniones de otros usuarios que les permitan mejorar la calidad del producto, así como su forma de programar. A pesar de reconocer las ventajas, su potencial y su interés, todavía no es un porcentaje elevado de estudiantes los que reconocen por su iniciativa crear proyectos con Scratch. Probablemente se deba al poco tiempo que llevan trabajando con el software y la escasa costumbre que poseen, en general, para realizar actividades que no formen parte obligatoria de su formación. Esto implica un cambio cultural en los estudiantes de la titulación de Informática de la Facultad de Filosofía, que debe ser afrontado paulatinamente.

Para finalizar, y a partir de la investigación realizada, queremos indicar dos recomendaciones que consideramos importantes en el caso de optar por la utilización de este tipo de herramientas para el desarrollo del pensamiento computacional. En primer lugar, mantener el uso de Scratch o herramientas similares para la elaboración de algoritmos como paso previo al aprendizaje de un lenguaje de programación, ya que facilita al estudiante el desarrollo del pensamiento computacional. En segundo lugar, motivar al estudiante a poner en práctica los aprendizajes en diseño de presentaciones, juegos y otros proyectos mediante este tipo de herramientas, de forma que sus habilidades de modelación, abstracción y creatividad se potencien para, posteriormente, utilizarlas en la resolución de problemas en cualquier actividad de orden académico o cotidiano.

Presentación del artículo: 27 de julio de 2015
Fecha de aprobación: 6 de septiembre de 2015
Fecha de publicación: 15 de septiembre de 2015

Pérez, H. O. y Roig-Vila, R., (2015). Entornos de programación no mediados simbólicamente para el desarrollo del pensamiento computacional. Una experiencia en la formación de profesores de Informática de la Universidad Central del Ecuador. *RED. Revista de Educación a Distancia, 46(9).* 15 de Septiembre de 2015. Consultado el (dd/mm/aa) en http://www.um.es/ead/red/46

Referencias

Anthony, N. y Soon, O. Y. (2015). *Development of a Novel Intelligent Social Game Design Learning Platform-'Gamewiz'.* Recuperado de: http://ircset.org/anand/2015papers/IRC-SET-2015_submission_38.pdf

Banchoff, C., Diáz, J., Queiruga, C., y Martín, E. (2014). *Experiencias de la Facultad de Informática en la Enseñanza de Programación en Escuelas con Software Libre*. Recuperado de: https://goo.gl/k96AVZ

Barr, D., Harrison, J. y Conery, L. (2011). *Computational Thinking: a Digital Age Skill for Everyone. Learning & Leading with Technology*, 38(6), 20-23. Recuperado de: http://eric.ed.gov/?id=EJ918910

Brennan, K. y Resnick, M. (2012). *Entrevistas basadas en artefactos para estudiar el desarrollo del Pensamiento Computacional (PC) en el diseño de medios interactivos*. Documento presentado en el encuentro anual de la "American Educational Research Association", AERA 2012, Vancouver, BC, Canada.

Bustillo, J., Vizcarra, M., & Aristizabal, P. (2014). *Análisis del proceso formativo de un grupo de reclusos en un taller de Scratch. RELATEC*, 13(1), 37.

Carralero Colmenar, N. (2011). *Entornos para enseñar Programación en Secundaria. Nuevos Enfoques. QUADERNS digitales.NET*. Recuperado de: http://goo.gl/nFAysj

Ferreira, A. y Rojo, G. (2006*). Enseñanza de la programación. TE&ET. Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, 1(1). Recuperado de: http://teyet-revista.info.unlp.edu.ar/files/No1/09_Ensenanza_de_la_programacion.pdf

Guerra, (2008). *Estrategias de Aprendizaje Colaborativo utilizando las Nuevas Tecnologías de Información y Comunicación. (Evaluación por Grupos). Revista Docencia Universitaria. Volumen IX. N° 2.* Universidad de Carabobo: Venezuela. Recuperado de: http://www.ucv.ve/fileadmin/user_upload/sadpro/Documentos/docencia_vol9_n2_2008/4_art._1Laura_Guerra.pdf

Jin, W. y Zhaohui, W. (2015). *A study of the experiment teaching of 'the Fundamentals of Computer' based on Computational Thinking*. En S. Yingying, C. Guiran y L.

Zhen (eds.), *Proceedings of the 2015 International Conference on Education, Management, Information and Medicine* (pp. 246-251). doi:10.2991/emim-15.2015.48

Larrea, E. (2013*). El currículo de la Educación Superior desde la complejidad sistémica*. Quito: ES.

López, J. C. (2014). *Actividades de aula con Scratch que favorecen el uso del pensamiento algorítmico*. Recuperado de: http://www.eduteka.org/pdfdir/tesis-juan-carlos-lopez.pdf

Olabe, J. C., Olabe, M. A., Basogain, X., Maiz, I., Castaño, C. (2011). *Programming and Robotics with Scratch in Primary Education.* En A. Méndez-Vilas (Ed.) *Education in a Technological World: Communicating Current and Emerging Research and Technological Efforts* (pp. 356–363).

Ospina Mejía, O. (2012). Comunidad Scratch: red social creativa para niños y jóvenes. Recuperado de: http://colombiadigital.net/actualidad/noticias/item/1588-comunidad-scratch-red-social-creativa-para-ninos-y-jovenes.html

Papert, S. (1982). *Desafío a la mente.* Buenos Aires: Ediciones Galápago.

Park, S.-Y., Song, K.-S. y Kim, S.-H. (2015). EEG *Analysis for Computational Thinking based Education Effect on the Learners' Cognitive Load.* En WSEAS *Recent Advances in Computer Science* (pp. 38-43). Recuperado de: http://www.wseas.us/e-library/conferences/2015/Malaysia/COMP/COMP-04.pdf

Peñaherrera, L. (2012). *Uso de TIC en escuelas públicas de Ecuador: análisis, reflexiones y valoraciones. Edutec, Revista Electrónica de Tecnología Educativa*, 40, 1-16. Recuperado de: http://edutec.rediris.es/Revelec2/Revelec40/pdf/Edutec-e_n40_Penaherrera.pdf

Pérez, H. (2011). *Influencia del razonamiento lógico verbal en el rendimiento académico de Programación.* (Tesis de maestría inédita). Quito: Universidad Central del Ecuador.

Resnick, M., (2007). *Sembrando las semillas para una sociedad más creativa. Laboratorio de medios de MIT, Massachussets.* Recuperado de http://www.eduteka.org/ScratchResnickCreatividad.php

Señas, P. y Moroni, N.(2002). *Herramientas no convencionales para el aprendizaje de la programación.* Bahía Blanca: Universidad Nacional del Sur.

Simari, G. (2011). *Los fundamentos computacionales como parte de las ciencias básicas en las terminales de la disciplina Informática*. Bahía Blanca: Universidad Nacional del Sur. Recuperado de: http://sedici.unlp.edu.ar/bitstream/handle/10915/27579/Documento_completo.pdf?sequence=1

Soler, Y. y Lezcano, M. (2009*). Consideraciones sobre la tecnología educativa en el proceso de enseñanza-aprendizaje. Una experiencia en la asignatura Estructura de Datos. Revista Iberoamericana de Educación*, 49/2, 1-9. Recuperado de: http://www.rieoei.org/expe/2863Soler.pdf

Taborda, D, y Medina, H. (2013). *Programación de computadores y desarrollo de habilidades de pensamiento en niños escolares: fase exploratoria.* Recuperado de: http://www.eduteka.org/pdfdir/Icesi_Investigacion_Scratch_FaseI.pdf

Vega, A. y Espinel, A. (2010). *Aspectos fundamentales para la enseñanza de programación básica en Ingeniería. Revista Avances en Sistemas e Informática*, 7(1), 7-14. Recuperado de: http://www.revistas.unal.edu.co/index.php/avances/article/view/20584/21624

Voogt, J., Fisser, P., Good, J., Mishra, P. Yadav, A. (2015). *Computational thinking in compulsory education: Towards an agenda for research and practice*. *Education and Information Technologies.* DOI: 10.1007/s10639-015-9412-6

Wing, J. (2009a). *Computational Thinking*. CACM viewpoint, vol. 49(3), pp. 33-35.

Wing, J. (2009b). *Computational Thinking and Thinking About Computing*. Traducción de Cubillán, J (2011). Recuperado de https://goo.gl/3CMUHc

Zapata, M. (2014). Pensamiento computacional y alfabetización digital (I). Recuperado de http://red.hypotheses.org/776

Zapotecatl, J. L. (2014). *Pensamiento Computacional.* Puebla: Instituto Nacional de Astrofísica, Óptica y Electrónica. Recuperado de: http://www.pensamientocomputacional.org/Files/pensamientocomputacional.pdf

Zappa, F (s.f.). *Frases y Citas Célebres de Frank Zappa.* Recuperado de http://akifrases.com/autor/frank-zappa

ANEXO 1

UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE FILOSOFÍA, LETRAS Y CIENCIAS DE LA EDUCACIÓN

CARRERA INFORMATICA

ENCUESTA DIRIGIDA A ESTUDIANTES

DATOS INFORMATIVOS

EDAD _____ GENERO _____

El objetivo de esta encuesta es recopilar información sobre el uso de Scratch como herramienta didáctica en la enseñanza de programación I.

INTRUCCIONES

a. Lea detenidamente las preguntas y marque con una X en la casilla seleccionada; escoja solamente una de las opciones presente.

(4) SIEMPRE (3) CASI SIEMPRE (2) A VECES (1) CASI NUNCA (0) NUNCA

No	ITEMS	SIEMPRE	CASI SIEMPRE	AVECES	CASI NUNCA	NUNCA
		4	3	2	1	0
1	¿La interface que presenta Scratch facilita la elaboración de proyectos o programas?					
2	¿Considera que la interface de Scratch es complicada para crear programas o proyectos?					
3	¿Considera que Scratch entrega recursos variados para crear proyectos o programas?					
4	¿Las herramientas que entrega Scratch son suficientes para elaborar proyectos o programas?					

5	¿Utiliza Scratch para elaborar proyectos o programas educativos?					
6	¿Utiliza Scratch continuamente en las clases de programación?					
7	¿El trabajo colaborativo en Scratch ha permitido mejorar su aprendizaje?					
8	¿El trabajo colaborativo aumenta el interés de los estudiantes para crear proyectos o programas en Scratch?					
9	¿Diseñar proyectos en Scratch promueve la creatividad?					
10	¿Considera que su imaginación se potencia al crear proyectos en Scratch?					
11	¿Realiza proyectos o programas en Scratch utilizando operadores lógicos y matemáticos?					
12	¿Diseñar proyectos en Scratch promueve el desarrollo del pensamiento?					
13	¿Utiliza Scratch para crear juegos educativos, que fomente el uso de la aplicación?					
14	¿Utiliza Scratch para realizar presentaciones para exposiciones en clase?					
15	¿Elabora proyectos innovadores o novedosos en Scratch, que permitan desarrollar sus habilidades en programación?					
16	¿Diseñar proyectos o programas en Scratch permite mejorar el aprendizaje de programación?					
17	¿El docente utiliza metodologías activas en las clases de programación?					
18	¿El docente fomenta la participación activa de los estudiantes?					
19	¿El docente es el único participe durante las clases de Programación?					
20	¿El docente incentiva el trabajo individual en las clases de programación?					
21	¿Considera que el trabajo individual permite adquirir nuevos conocimientos?					
22	¿El docente fomenta el trabajo en equipo utilizando metodologías participativas?					
23	¿El docente realiza trabajos grupales en clases permitiendo la participación de los estudiantes?					